REMARKS

Reconsideration of this application, as amended, is respectfully requested.

Claims 1-30 are pending. Claims 1-3, 6-8, 11-13, 16-18, 21-23 and 26-28 stand rejected. Claims 4, 5, 9, 10, 14, 15, 19, 20, 24, 25, 29 and 30 have been objected to.

Claims 1, 5, 6, 10, 11, 15, 16, 20, 21, 25, 26, and 30 have been amended. Claims 4, 9, 14, 19, 24, and 29 have been cancelled. Support for the amendments is found in the specification, the drawings, and in the claims as originally filed. Applicants submit that the amendments do not add new matter.

Rejections Under 35 U.S.C. § 102(e)

Claims 1-3, 6-8, 11-13, 16-18, 21-23 and 26-28 stand rejected under 35 U.S.C. § 102(e) as being anticipated by Cohen et al. U.S. Patent No. 6,718,539 ("Cohen"). The Examiner stated that

> As to claims 1, 6 ,11, 16 , 21and 26, Cohen teaches inserting at least one null operation instruction (col 15, lines 40-45) in an instruction set (col 15, lines 40-67); and recording information within a data field (i.e. a NOP by definition has bit fields that contains/records/stores data) of the null operation instruction (col 15, lines 40-67).

(p. 2, Office Action 7/13/04)

Applicants respectfully submit that claim 1, as amended is not anticipated by Cohen under 35 U.S.C. § 102(e). Amended claim 1 includes the following limitations:

> A method comprising:
> inserting at least one null operation instruction in an instruction set; and
> <u>recording live reference information for a garbage collection process within a data field of the null operation instruction</u>.

(Amended claim 1) (emphasis added)

In contrast, Cohen does not disclose the limitation of recording live reference information for a garbage collection process within a data field of the null operation instruction.

Cohen discloses that

In one example, the circuit 100 may be configured to present a wait signal to stop the machine. Alternatively, the circuit 100 may be configured to insert a number of null operation (NOP) instructions as the first instructions presented to the CPU 102.

When the circuit 100 (i) needs the support of the JVM, (ii) is to return control to the JVM, or (iii) because of an interrupt some other code is to be executed, the circuit 100 may be configured to recognize that the address presented by the CPU 102 is not within the predefined memory space and stop executing.

(Cohen Col. 15, lines 40-45). Cohen also discloses that

In one example, the circuit 100 may be configured to present a wait signal to stop the machine. Alternatively, the circuit 100 may be configured to insert a number of null operation (NOP) instructions as the first instructions presented to the CPU 102.

When the circuit 100 (i) needs the support of the JVM, (ii) is to return control to the JVM, or (iii) because of an interrupt some other code is to be executed, the circuit 100 may be configured to recognize that the address presented by the CPU 102 is not within the predefined memory space and stop executing. When the circuit 100 stops executing, the paths between the CPU 102 and memory system 104 are generally re-opened.

Referring to FIG. 9a, a diagram illustrating the circuit 100 entering the program memory space of FIG. 8 in terms of cycles is shown. When the JVM decides to translate instruction codes (e.g., JAVA bytecodes) into a sequence of native instruction codes, the first address of the JAVA bytecodes (e.g., Y) may be placed in the BCPC register of the circuit 100. The CPU 102 then may be instructed to execute a jump (e.g., JSUB or JMP) to JOD_START0. When the processor executes the jump, the address JOD_START0 generally appears on the processor address bus. The circuit 100 may be configured to detect the presence of the address JOD_START0 on the processor address bus and begin translating instructions (e.g., indicated by the Y in parentheses) located starting at the address stored in the BCPC register.

The circuit 100 may present a number of null operation (NOP) instructions to the CPU 102 while the pipeline of the circuit 100 begins filling

(Cohen Col. 15, lines 40-67). Cohen also discloses that

The present invention may comprise a small hardware block tightly coupled to a processor (e.g., MIPS, ARM, 68K, etc.). The hardware block and appropriate software generally turn the processor into a fast Java virtual machine (JVM). Basic tasks such as stack and program counter management may be performed with no penalty at all by the hardware block while the processor may be performing the operations required by the Java bytecode instructions.

The present invention may represent a more suitable compromise between speed, memory requirements and compatibility than provided by conventional solutions. The present invention may provide the ability to easily run native code and Java bytecode on

> the same processor with reduced overload. For example, the present invention may provide a clear and simple implementation path for all major real-time operating systems (RTOS) and other operating systems (OSs) available (e.g., pSOS, Microsoft WindowsCE.RTM., etc.). The present invention may be configured to execute legacy code. For example, the hardware portion may be configured to operate with all popular processors (e.g., MIPS, ARM, Motorola 68K, etc.). The present invention may provide a performance boost better than the JIT boost.
>
> The present invention may provide a low cost solution that may use a small silicon area (e.g., about 20-30 Kgates) and a minimal memory foot print (e.g., a general increase in memory requirements of not more than 5%).

(Cohen Col 3 Lines 25-50)

Cohen does not disclose that the information recorded in the data field of the null operation instruction is live information for a garbage collection process as identified by the Examiner.

For this reason applicants respectfully submit that claim 1 is not anticipated by Cohen. Given that claims 2, 3, and 5, depend, directly or indirectly from claim 1, applicants respectfully submit that claims 2, 3, and 5, are, likewise, not anticipated by Cohen.

Further, given that claims 6, 11, 16, 21, and 26, as amended, include the limitation of recording live reference information for a garbage collection process within a data field of the null operation instruction, applicants respectfully submit that claims 6, 11, 16, 21, and 26 are not anticipated by Cohen. Given that claims 7, 8, and 10, claims 12, 13, and 15, claims 17, 18, and 20, claims 22, 23, and 25, and claims 27, 28, and 30, depend, directly or indirectly from claims 6, 11, 16, 21, and 26, respectively, applicants respectfully submit that claims 7, 8, 10, 12, 13, 15, 17, 18, 20, 22, 23, 25, 27, 28, and 30, are, likewise, not anticipated by Cohen.

It is respectfully submitted that in view of the amendments and arguments set forth herein, the applicable rejections and objections have been overcome. If there are any additional charges, please charge Deposit Account No. 02-2666 for any fee deficiency that may be due.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 10\6\04 By: [illegible]
Tom Van Zandt
Reg. No. 43,219

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(408) 720-8300